

MACHINE

LEARNING

MACHINE LEARNING ES LA RAMA DE LA COMPUTACIÓN QUE SE ENCARGA DE CONSTRUIR ALGORITMOS QUE APRENDEN A HACER ALGO ÚTIL A PARTIR DE LOS DATOS.

LOS ALGORITMOS DE MACHINE LEARNING SE DIVIDE EN DOS RAMAS:

APRENDIZAJE SUPERVISADO

ESTE PERMITE REALIZAR PREDICCIONES BASADAS EN LAS CARACTERÍSTICAS ANALIZADAS EN DATOS. TENEMOS LABELS.

REGRESIÓN

QUEBEMOS PREDDECIR EL VALOR DE UNA VARIABLE NUMÉRICA Y CONTINUA. EN GENERAL CONTAMOS CON UN SET DE ENTRENAMIENTO EN EL CUAL CONOCEMOS EL VALOR DE LA VARIABLE QUE QUEDEMOS PREDDECIR. EL CASO MÁS SIMPLE ES LA REGRESIÓN LINEAL.

CLASIFICACIÓN

QUEBEMOS PREDDECIR LA CATEGORÍA A LA QUE PERTENECE NUESTRO DATO. FRECUENTEMENTE TIENE POCOS VALORES POSIBLES Y EN MUCHOS CASOS LOS VALORES SON DOS (CLASIF. BINARIA). TAMBién CONTAMOS CON UN SET DE ENTRENAMIENTO EN EL CUAL PARA CADA DATO CONOCEMOS LA CLASE A LA CUAL PERTENECE.

ANALISIS DE SENTIMIENTO

ESTE ES UN CASO TÍPICO DE CLASIFICACIÓN BINARIA. ACA QUEDEMOS SABER SI UN CIERTO TEXTO ES POSITIVO O

NEGATIVO, ES DECIR SI HABLA BIEN O MAL DE UN CIERTO TEMA. UNA APLICACIÓN MUY CONOCIDA ES SI LOS REVIEWS DE UN PRODUCTO SON BUENOS O MALOS!



APRENDIZAJE NO SUPERVISADO

NOS PERMITE HACER PREDICCIONES SIN TENER DATOS ETIQUETADOS.
NO TENEMOS LABELS

CLUSTERING CONTAMOS CON DATOS QUE QUEDÉMOS DIVIDIR DE FORMA AUTOMÁTICA. EN ALGUNOS CASOS, LA CANTIDAD DE CLUSTERS LA DEBEMOS INDICAR PREVIAMENTE Y EN OTROS ALGORITMOS ES CAPAZ DE DETERMINARLO POR SI MISMO. PARA ESTO NO NECESITAMOS CONOCER EL VALOR DE UNA VARIABLE O CLASE PARA CADA PUNTO; SOLO SE NECESITAN LOS DATOS EN CRUDO.

DETECTOR ANOMALIAS LA DETECCIÓN DE ANOMALIAS IMPLICA EL RECONOCIMIENTO Y COSECHÓN O ELIMINACIÓN DE DATOS ERRÓNEOS. ES ERROREO SI TIENE VALORES IMPOSIBLES PARA UNO O MAS ATRIBUTOS. DEL MODELO, CIERTOS ALGORITMOS SON MÁS SENSIBLES.

SET DE DATOS



EL SET DE DATOS TIENE M FILAS (OBSERVACIONES) Y N COLUMNAS (FEATURES O VARIABLES).

FEATURE ENGINEERING

ES EL PROCESO DE CREACIÓN DE FEATURES. ESTA TAREA ES MAYORMENTE MANUAL Y ES UN PROCESO FUNDAMENTAL PARA EL FUNCIONAMIENTO DE LOS ALGORITMOS DE MACHINE LEARNING.

① NUEVOS FEATURES INDIVIDUALES

PODEMOS CREAR NUEVOS FEATURES:

- **BASADOS EN TIEMPO** (POR EJ. EL \$ DE UN PRODUCTO HACE UN MES ATRÁS, CANTIDAD DE VENTAS EN..)
- **ESTADÍSTICOS** (EJ. EL \$ MAX, MIN, PROM...)
- **BASADOS EN KNN** (EJ. PROMEDIO DE X PARA LOS K VECINOS MÁS CERCANOS)

- BASADOS EN EL TEXTO (EJ PODEMOS SACAR INFO DEL TEXTO SABIENDO SI CONTIENE O NO O ALGO, TMB APPLICABLE TF-IDF)

② TRANSFORMACIÓN DE FEATURES

- LOGARITMICA : NOS PERMITE CONVERTIR DATOS QUE TIENEN DISTRIBUCIÓN EXP EN NORMAL
- MATEMÁTICAS : YA SEA LA RAÍZ CUAD. LA INVERSA, EL SENO ..
- NORMALIZACIÓN : RESTAR A LA VARIABLE SU PROMEDIO, ESTO MISMO DIVIDIR POR STD O POR RANGO
- BINNING : CONVIERTE UNA VARIABLE NÚMÉRICA EN CATEGÓRICA (POR EJ. $X > 20 = 0$ $X < 20 = 1$) ESTA ES ÚTIL PARA ARBOLES.

TODAS ESTAS SON MUY ÚTILES
Y POPULARES EN DEDICADAS
NEURONALES.

③ ENCODING DE VARIABLES CATEGÓRICAS.

- ONE HOT ENCODING : ESTE MÉTODO SIRVE CUANDO MI MODELO DE ML NO SOPORTA VARIABLES CATEGÓRICAS, ENTONCES LO QUE HACEMOS ES CREAR UNA COLUMNA POR CADA VALOR POSIBLE QUE TIENE LA VARIABLE. EL PROBLEMA CON ESTE MÉTODO ES QUE SI HAY MUCHOS VALORES GENERAMOS MUCHAS COLUMNAS Y ESO NO ESTA BUENO.

UNA SOLUCIÓN AL ONE HOT ENCODING ES EL BINARY ENCODING. AHORA CADA COLUMNA REPRESENTA LA CANTIDAD DE BITS QUE UTILIZAMOS PARA CODIFICAR LOS VALORES. ES MUCHO MÁS EFICIENTE PORQUE DEDUZCO LA CANTIDAD EN $\log_2 x$.

Ciudad	Gasto	Ad
Moscu	100	0
Moscu	20	1
Paris	105	1
Moscu	50	0
Roma	120	0
Paris	40	0
Roma	80	0
Londres	50	1



Gasto	Ad	Moscu	Paris	Roma	Londres
100	0	1	0	0	0
20	1	1	0	0	0
105	1	0	1	0	0
50	0	1	0	0	0
120	0	0	0	1	0
40	0	0	1	0	0
80	0	0	0	1	0
50	1	0	0	0	1

Ciudad	Gasto	Ad
Moscu	100	0
Moscu	20	1
Paris	105	1
Moscu	50	0
Roma	120	0
Paris	40	0
Roma	80	0
Londres	50	1



Gasto	Ad	C2	C1	C0
100	0	0	0	1
20	1	0	0	1
105	1	0	1	0
50	0	0	0	1
120	0	0	1	1
40	0	0	1	0
80	0	0	0	1
50	1	1	0	0

Moscu = 001
Paris = 010
Roma = 011
Londres = 100

MEAN ENCODING: LA IDEA ES CODIFICAR LAS VARIABLES CATEGÓRICAS EN BASE A LOS LABELS. HAY QUE TENER CUIDADO YA QUE SE PUEDE FILTRAR INFORMACIÓN DE LOS LABELS A LOS FEATURES DE ENTRENAMIENTO, Y GENERAR OVERFITTING.

Ciudad	Gasto	Ad
Moscu	100	0
Moscu	20	1
Paris	105	1
Moscu	50	0
Roma	120	0
Paris	40	0
Roma	80	0
Londres	50	1



Gasto	Ad	Ciudad-Mean
100	0	0.33
20	1	0.33
105	1	0.5
50	0	0.33
120	0	0
40	0	0.5
80	0	0
50	1	1

Peligro: se filtra información de los labels a los features de entrenamiento.

El cálculo no debe utilizar los datos de test.

REEMPLAZO LA COLUMNA CATEGÓRICA POR EL PROMEDIO DE CUANTOS UNOS HAY SOBRE LAS APARIACIONES DEL LABEL. ESTO ESTABLECE UN ORDEN ENTRE LAS VARIABLES, Y SOLO GENERAMOS UNA ÚNICA COLUMNA.

$$\frac{P}{P+N} \rightarrow \text{ESTA ES LA}$$

$$\ln\left(\frac{P}{N}\right) * 100$$

$$\sum(P)$$

$$P - N$$

P= Positivos, N= Negativos (para variables binarias)

FORMULAS POSIBLES DE MEAN ENCODING, QUE PODRÍAN AYUDAR A REDUCIR EL OVERFITTING.

LAS SIGUIENTES SON ALGUNAS SOLUCIONES PARA QUE NO SE FILTRE INFORMACIÓN DE LOS LABELS

CV MEAN ENCODING

Feature	Feature-Mean	Target
Moscu	0.5	0
Moscu	0.25	1
Moscu	0.25	1
Moscu	0.5	0
Moscu	0.5	0

(usando LOOCV, puede usarse cualquier esquema K-Fold)

SE USA REGISTRO A REGISTRO

SMOOTHING

ES UN HIPERPARÁMETRO.

$$\frac{\text{mean(target)} * \text{nrows} + \text{globalmean} * \alpha}{\text{nrows} + \alpha}$$

Alfa controla la cantidad de regularización a usar

ADDING NOISE: EL RUIDO DEGRADA LA CALIDAD DEL ENCODING

EXPANDING MEAN: SE PUEDE UTILIZAR SI TENEMOS UN FEATURE TEMPORAL. EL DF TIENE QUE ESTAR ORDENADO POR TIEMPO

④ INTERACCIÓN ENTRE FEATURES

● CATEGÓRICOS

ESTOS SON MUY IMPORTANTES CUANDO UTILIZAMOS MODELOS BASADOS EN ARBOLES, PORQUE ESTOS NO PUEDEN GENERAR LA INTERACCIÓN ENTRE FEATURES. ADEMÁS, SOLO HACE SPLIT POR UN FEATURE O POR EL OTRO Y ESTA ELECCIÓN ES POB GREEDY, NO SABE EL OPTIMO (\$ Y CANT VENTAS)

- CREAR NUEVOS FEATURES QUE SEAN LA CONCATENACIÓN DE 2 FEATURES Y ENCODEAR ESTE NUEVO FEATURE
- ENCODEAR CADA FEATURE INDIVIDUALMENTE Y LUEGO GENERAR EL PRODUCTO ENTRE LOS ENCODINGS.

● NÚMERICAS

- MULTIPLICACIÓN
- SUMA
- DIFERENCIA
- DIVISIÓN → RATIOS: RECORDAD QUE MÁS OBS MÁS SEGUNDOS ESTAMOS

$$1/2 \neq \frac{250}{500}$$

- GENERAR INTERVALOS DE CONFIANZA

CON N VARIABLES HAY $N * N$ INTERACCIONES POSIBLES, SI TOMAMOS DE A DOS, SOLO DEBEMOS SELECCIONAR LOS QUE SON REALMENTE UTILES.

- ENTONCES A PARTIR DEL SET DE DATOS QUE DEMOS ENTRENAR UN MODELO DE ML QUE NOS PERMITA PREDICIR EL LABEL A PARTIR DE LOS FEATURES.

VALIDACIÓN DEL MODELO

PARA EVALUAR EL MODELO ENTRENADO HAY QUE HACER PREDICCIONES CON DATOS PARA LOS CUALES CONOCEMOS

EL VALOR A PREDDECIR (LABEL). EN GENERAL SE DIVIDE AL SET DE DATOS EN DOS **TRAINING SET** Y **TEST SET**. ENTONCES EL TRAINING SET LO USAMOS PARA ENTRENAR EL MODELO Y EL TEST SET LO USAMOS PARA MEDIR LA PERFORMANCE DEL MODELO, ES DECIR, SI GENERALIZÓ BIEN.



EL SPLIT EN ALGUNOS CASOS PODEMOS HACERLO AL AZAÑO (80% TRAIN - 20% TEST). EN OTROS CASOS ES IMPORTANTE HACERLO POR TIEMPO PARA EVITAR TIME-TRAVELLING. (TRAIN - DATOS ABRIL) TEST - DATOS MAYO)

SETS DESBALANCEADOS

SI EL SET DE DATOS ESTÁ MUY DESBALANCEADO PUEDE SER DIFÍCIL ENTRENAR EL MODELO. LAS POSIBLES SOLUCIONES

- OVERSAMPLEAR LA CLASE MINORITARIA.
- SUBSAMPLEAR LA CLASE MAYORITARIA.
- MANEJAR EL DESBALANCEO CON HIPER-PARAMETROS DEL MODELO.

UNA VEZ QUE TENEMOS LOS SETS CORRECTAMENTE Y ENTRENAMOS EL MODELO DEBEMOS UTILIZAR ALGUNA METRÍCA PARA EVALUAR LA PERFORMANCE.

METRICAS

ESTAS TIENEN QUE TENER SENTIDO PARA EL PROBLEMA QUE QUEDEMOS RESOLVER

- ACCURACY
- PRECISIÓN

- RECALL
- RMSE

¿QUE ES ENTRENAR?

CADA MODELO DE ML TIENE UN CONJUNTO DE PARAMETROS E HIPER-PARAMETROS QUE NECESITA PARA FUNCIONAR.

PARAMETROS

ESTOS LOS DESCUBRE EL ALGORITMO A PARTIR DE LOS DATOS.

HIPER-PARAMETROS

ENTRENAMIENTO DEL MODELO. ESTOS SON VALORES QUE DEBEMOS INDICAR POR AFUERA, NO SON PARTE DEL PROCESO DE OPTIMIZACIÓN DEL MODELO.

→ ENTONCES EL ENTRENAMIENTO DE UN MODELO ES UN PROBLEMA DE OPTIMIZACIÓN. BUSCAMOS LOS PARAMETROS OPTIMOS PARA MINIMIZAR/MAXIMIZAR LA METRICA ELEGIDA. DEPENDIENDO DEL MODELO VARIÁ LA TÉCNICA DE OPTIMIZACIÓN A USAR.

NECESITAMOS ENCONTRAR EL CONJUNTO ÓPTIMO DE HIPER-PARAMETROS PARA LUEGO ENCONTRAR EL MODELO ÓPTIMO. A LA BUSQUEDA DE HIPER-PARAMETROS SE LA CONOCE COMO

TUNING

① GRID-SEARCH

ESTABLECEMOS UN CONJUNTO DE VALORES A PROBAR POR CADA HIPER-PARAMETRO. LUEGO SE PRUEBA CADA COMBINACIÓN DE HIPER-PARAM Y NOS QUEDAMOS CON LA MEJOR.

UNA VEZ QUE ENCONTRAMOS LOS VALORES OPTIMOS PODEMOS REFINAR LA BUSQUEDA, POR EJ SI EL OPTIMO ESTA ENTRE DOS VALORES PROBAMOS MAS RESOLUCIÓN

LA DESVENTAJA QUE PUEDE PRESENTAR EN ESTE ALGORITMO ES QUE CONSUME MUCHO TIEMPO YA QUE TIENE QUE EVALUAR TODAS LAS COMBINACIONES POSIBLES. ALGUNOS MODELOS TIENEN MUCHOS HIPEZ - PARAMETROS.

② RANDOM SEARCH

DEFINIMOS UN CONJUNTO DE VALORES POSIBLES POR CADA HIPEZ PARAMETRO. Luego PROBAMOS 'k' COMBINACIONES ALEATORIAS y nos quedamos con la mejor. EN ESTE ALGORITMO DECIDIMOS CUANTO TIEMPO INVERTIR PERO NO PROBAMOS TODAS LAS COMBINACIONES, NOS PODEMOS ESTAR SALTANDO LA SOLUCIÓN OPTIMA.

③ BÚSQUEDA BAYESIANA

DEFINIMOS UNA DISTRIBUCIÓN DE PROBABILIDADES Y BANGOS PARA CADA HIPEZ - PARAMETRO. EL ALGORITMO VA PROBANDO COMBINACIONES DE FORMA BAYESIANA, ES DECIR, SE VA MOVIENDO DEPENDIENDO SI ES MEJOR O NO DEPENDIENDO DE LAS PROBABILIDADES.

PARA ESTOS 3 ALGORITMOS TENEMOS QUE PROBAR UN CONJUNTO DE HIPEZ - PARAMETROS. PARA ESTO NO PODEMOS USAR EL TEST SET, TAMPOCO PODEMOS PROBAR CON EL SET DE ENTRENAMIENTO YA QUE LOS H-P SERIAN OPTIMOS PARA ESE SET EN PARTICULAR, NO PARA LOS DATOS POR AFUERA DEL MISMO.

¿COMO HACEMOS?

SET DE VALIDACIÓN

CON ESTO PODEMOS PROBAR LOS H-P
LA IDEA ES DIVIDIR EL TRAIN-SET EN DOS : UNO DE ENTRENAMIENTO Y EL OTRO DE VALIDACIÓN, PERO SI EL SET DE VAL ES FIJO, LOS H-P SOLO SON OPTIMOS PARA ESE CONJUNTO. ENTONCES, UTILIZAMOS :

CROSS-VALIDATION

ESTO DIVIDE EL SET DE USAMOS CADA FOLD COMO SET DE TRAIN EN K FOLDS, NANDO CON EL RESTO. LUEGO PROMEDIAMOS LA METRICA EN K VALIDACIONES.

UNDERFITTING

- SE PRODUCE CUANDO TENEMOS UN ERROR DE ENTRENAMIENTO ALTO (ENTRENAR Y PREDICIR CON EL MISMO SET TRAIN)
- EL MODELO AJUSTA MAL AL SET DE ENTRENAMIENTO
- EL MODELO TIENE 'VISION BOBOSA'
- NO TIENE SUFFICIENTE CAPACIDAD EXPRESIVA
- NO TIENE SUFFICIENTE COMPLEJIDAD O GRADOS DE LIBERTAD.

SOLUCIÓN :

CAMBiar A UN MODELO MÁS COMPLEJO, MAS EXPRESIVO.

OVERFITTING

- EL MODELO TIENE MUY BUEN RESULTADO PARA EL SET DE ENTRENAMIENTO PERO MALO PARA EL SET DE TEST
- GENERALIZA MAL Y ALUCINA. (EJ SI EL DATO CONTIENE UNA DESCRIPCION Y DICE 'HOLA' LA CLASIFICA SIEMPRE DE LO MISMO, SIN TENER EN CUENTA LOS DEMAS FEATURES).
- ES DEMASIADO EXPRESIVO.
- SI EL MODELO ES MUY COMPLEJO PUEDE HABER MEMORIZADO EL SET DE ENTRENAMIENTO.
- UN MODELO CON MUCHOS GRADOS DE LIBERTAD PUEDE TENER DEMASIAS SOLUCIONES POSIBLES.

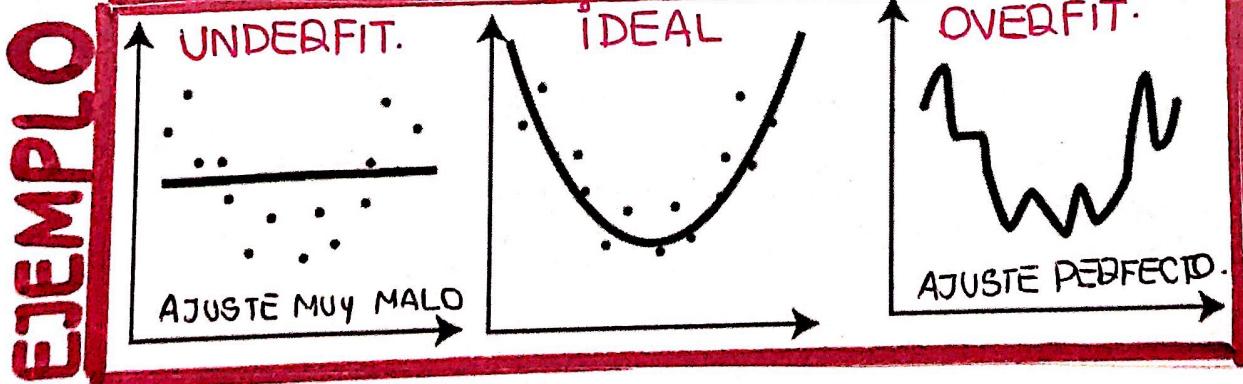
SOLUCIÓN : CONSEGUIDZ MAS DATOS ó DISMINUIDZ LA COMPLEJIDAD DEL MODELO.

TMB SE PUEDE USAR

REGULARIZACIÓN

ESTA ES UNA TECNICA POR LA CUAL PENALIZAMOS A UN MODELO DE ML EN FUNCION DE SU COMPLEJIDAD.

PQOB. DE REGRESIÓN

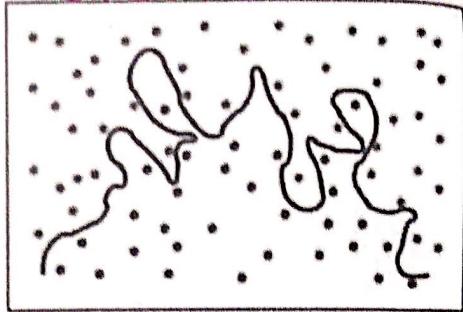
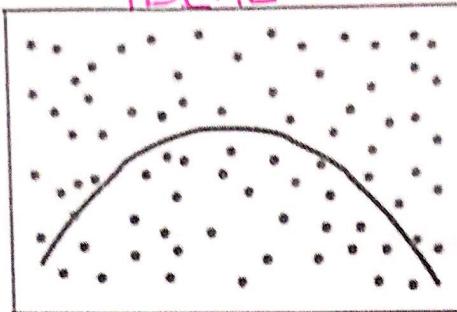
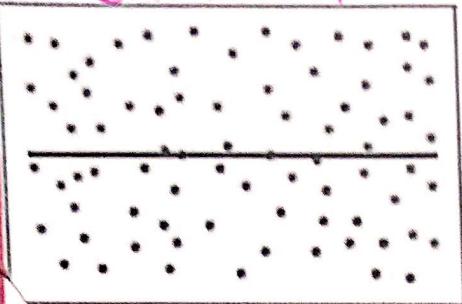


PROB. DE CLASIFICACIÓN

UNDERFIT.

IDEAL

OVERFIT.



SON PUNTOS ROJOS Y AZULES.

BIAS Y VARIANCE

ES OTRA FORMA DE VER EL PROBLEMA DE OVERFIT Y UNDERFIT.

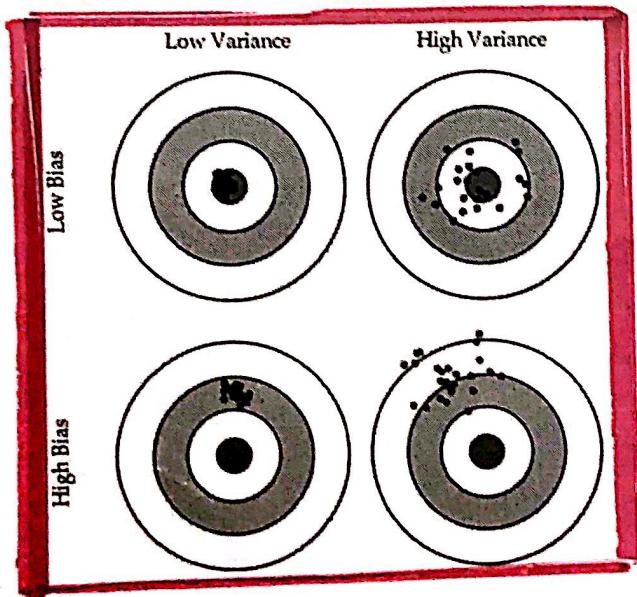
CUANTO APROXIMA LA ESTIMACIÓN AL SET DE ENTRENAMIENTO.

ASOCIADO AL ERRORE QUE TENEMOS EN EL SET DE ENTRENAMIENTO.



CUANTOS GRADOS DE LIBERTAD TIENE EL RESULTADO DE LA ESTIMACIÓN · ASOCIADO AL ERROR QUE TENEMOS EN EL SET TEST

	LOW BIAS	HIGH BIAS
LOW VARIANCE	GOOD !!	UNDERFITTING
HIGH VARIANCE	OVERFITTING	BAD !!



ENSAMBLÉS

LOS MEJORES ALGORITMOS DE ML SUELEN SUGERIR DE LA COMBINACIÓN DE VARIOS ALGORITMOS. ESTO NO QUIERE DECIR QUE EN LA PRACTICA SIEMPRE SEA BUENA IDEA USAR UN ENSAMBLE. EN ALGUNOS CASOS AGREGAMOS COMPLEJIDAD A CAMBIO DE UNA PEQUEÑA MEJORA EN LOS RESULTADOS. POR EJEMPLO USAR UN SOLO ALGORITMO QUE NO DA UNA PRECISIÓN DE 80,5%. O UN ENSAMBLE DE 200 ALGORITMOS PARA UNA PRECISIÓN DE 80,7%. PROBABILMENTE SEAN RESULTADOS EQUIVALENTES Y EL SEGUNDO ES MUCHO MÁS COMPLEJO.

BAGGING

♥ BAGGING IMPLICA APlicar EL MISMO CLASIFICADOR N VECES EN CONJUNTO CON BOOTSTRAPPING. BOOTSTRAPPING CONSISTE EN TOMAR UNA MUESTRA DEL TRAIN TEST CON REEMPLAZO DE IGUAL TAMAÑO QUE ESTE. (UN DATO PUEDE ESTAR VARIAS VECES EN LA MUESTRA). ENTONCES ENTRENAMOS NUESTRO CLASIFICADOR CON LOS DISTINTOS BOOTSTRAPS Y OBTENER N CLASIFICADORES DISTINTOS. LUEGO ESTOS LOS APlicAMOS AL SET ORIGINAL Y PROMEDIAMOS LOS RESULTADOS PARA UNA RTA FINAL.

- SIRVE PARA CLASIFICACIÓN Y REGRESIÓN
- DISMINUYE EL OVERFITTING, YA QUE NINGUNO DE LOS N CLASIFICADORES PUEDE SOBRE-AJUSTAR EL SET COMPLETO.

PARA CADA CLASIFICADOR EXISTE UN SET DE REGISTROS QUE QUEDA AFUERA : OUT OF BAG. LA PRECISIÓN DE UN CLASIFICADOR QUE USA BAGGING SE PUEDE OBTENER MEDIANTE LA CLASIFICACIÓN DE LOS REGISTROS OOB. ESTOS SIRVEN COMO SET DE VALIDACIÓN

Regí	Class	c1	c2	c3	Ave
0	0	0	0	0	0
1	1	0	0	0	0
2	0	0	0	0	0
3	1	1	1	0	1
4	0	1	0	0	1
5	1	1	0	0	1
6	0	1	1	0	1
7	0	0	0	0	0
8	1	1	1	1	1
9	1	1	1	0	1
P		7/10	7/10	6/10	8/10
OOB		2/4	3/4	0/3	0.42

= REGISTROS QUE FORMAN PARTE DEL BOOTSTRAP.
 = REGISTROS OOB

→ PRECISIÓN DEL CLASIFICADOR ENTREO

ENTONCES PARA BAGGING VAMOS A BUSCAR LOS H-P QUE NOS DEN MEJOR PROMEDIO DE PRECISIÓN OBB.

BOOSTING

ESTE MÉTODO CONSISTE EN CONSTRUIR UN ALGORITMO MUY PRECISO A PARTIR DE UN CONJUNTO DE ALGORITMOS MUY SIMPLES LOS CUALES POR SEPARADO PUEDEN FUNCIONAR MUY MAL.

EL MÉTODO CONSISTE EN ENTRENAR UN ALGORITMO SIMPLE, ANALIZAR SUS RESULTADOS Y LUEGO ENTRENAR OTRO ALGORITMO SIMPLE EN DONDE SE LE DA MAYOR PESO A LOS RESULTADOS PARA LOS CUALES EN EL ANTERIOR TUVO PEOR PERFORMANCE. A SU VEZ CADA ALGORITMO TIENE SU PESO PROPORCIONAL A LA CANTIDAD DE ACIERTOS QUE TUVO PARA EL TRAIN SET. EL RESULTADO FINAL ES EL PROMEDIO PONDERADO DE TODOS LOS ALGORITMOS USADOS CON SUS PESOS.

♥ ESTA TÉCNICA ES IMPORTANTE EN LOS ALGORITMOS DE TIPO GRADIENT BOOSTING COMO GBN O XGBOOST.

COMBINANDO ALGORITMOS ≠ MAJORITY VOTING

EN ESTE MODELO CADA CLASIFICADOR DA UN VOTO A CADA CLASE, LA CLASE QUE TIENE MAYORÍA DE VOTO SERÍA EL RESULTADO FINAL.

A MODO DE EJEMPLO SUPONGAMOS QUE LOS SIGUIENTES DATOS, TODOS SON CLASE 1. MOSTRAMOS 3 CLASIFICADORES DISTINTOS.

$$\begin{aligned}1111111100 &= 80\% \\1111111100 &= 80\% \\1011111100 &= 70\% \\\hline1111111100 &= 80\%\end{aligned}$$

$$\begin{aligned}1111111100 &= 80\% \\0111011101 &= 70\% \\1000101111 &= 60\% \\\hline1111111101 &= 90\%\end{aligned}$$

✓ LO QUE PODEMOS NOTAR ES QUE USANDO 3 CLASIFICADORES CUYAS PREDICCIONES ESTAN MUY CORRELACIONADAS ES DIFÍCIL OBTENER UNA MEJOR PREDICCIÓN. EN EL CASO CONTARIO EL RESULTADO MEJORA NOTABLEMENTE.

✓ A ESTO TAMBIEN SE LE PUEDE DAR UN PESO A CADA MODELO. PARA QUE TENGA MAYOR PESO EL QUE MEJORA FUNCIONA Y MENOR PESO LOS OTROS.

AVERAGING

- ✓ PROMEDIA EL RESULTADO DE VARIOS CLASIFICADORES
- ✓ SIRVE PARA REGRESIÓN Y CLASIFICACIÓN

Reg	c1	c1 rank	c2	c2 rank	Ave	Final Prob
1	0.57	3	0.3605	1	2	0.25
2	0.04	4	0.3502	3	3.5	1
3	0.96	2	0.35	4	3	0.75
4	0.99	1	0.36	2	1.5	0

$$\text{PROBABILIDAD FINAL} = \frac{x - \min}{\max - \min}$$

BLENDING

1. SEPARAR UN 10% DEL SET DE ENTRENAMIENTO.

2. Entrenar n clasificadores diferentes con el 90% del set de entrenamiento (separando este 90% en sets de entrenamiento y validación para optimizar cada modelo).
3. Realizar n predicciones para el 10% que separamos usando cada uno de los n clasificadores que entrenamos.
4. Entrenar un modelo *blender* que use las predicciones aprendidas para realizar la clasificación final.
5. Entrenar los n modelos con el set de entrenamiento completo.
6. Realizar predicciones con estos modelos para el set de test.
7. Combinar las predicciones para el set de test usando el modelo *blender*.
8. Combinar los sets de entrenamiento y test en un nuevo super-set de entrenamiento en donde los labels para lo que era el set de test son los que predijo el *blender*.
9. Separar un 10% del super-set.
10. Entrenar los n modelos en el 90% restante del super-set.
11. Predecir los resultados para el 10% que separamos usando cada modelo.
12. Entrenar un *blender* para combinar estas predicciones en la predicción final.
13. Entrenar los n modelos usando el super-set completo.
14. Aplicar los modelos aprendidos al set de test.
15. Aplicar el super-blender al set de test.

EJERCICIOS DE MACHINE LEARNING

- 1) IDENTIFICAR SI ES UN PROBLEMA DE CLASIFICACIÓN O REGRESIÓN.
- 2) ELEGIR QUE MODELO DE MACHINE LEARNING SE VA A UTILIZAR PARA RESOLVER EL PROBLEMA Y PORQUE.
- 3) COMO SE VA REALIZAR EL PROCESAMIENTO DE DATOS Y LUEGO EL FEATURE ENGINEERING.

ENTRENAMIENTO

- 4) REALIZAR UNA PARTICIÓN ALEATORIA DEL SET DE DATOS EN SET DE TRAIN Y SET DE TEST CON UNA PROPORCIÓN DE 15% Y 85% (PUEDE SER OTRA). INDICAR QUE FEATURE SE UTILIZA COMO TARGET/LABEL
- 5) SE ENTRENA EL MODELO CON EL SET DE TRAIN Y SUS RESPECTIVOS LABELS/TARGET

PREDICCIÓN

- 6) SE REALIZAN LAS PREDICCIONES CON EL SET DE TEST Y SE EVALUAN CON LAS METRÍCAS ELEGIDAS.

OPTIMIZACIÓN

- 7) EL PRIMER INTENTO TRABAJA CON LOS HIPO - PARAM POR DEFECTO, SEGUARDAMENTE NO SEAN TAN BUENOS RESULTADOS, ASI QUE PARA MEJORARLOS SE PUEDE REALIZAR UN TUNING.

- 8) FINALMENTE REALIZAMOS LAS PREDICCIONES DE NUESTROS DATOS(QUE NO SABEMOS EL T) CON EL MODELO YA ENTRENADO.